Toolbox 1 Report - Antoine Assaf

**Background**

PhiFlow is a physics-based library for simulating fluid dynamics, aerodynamics, and other PDE-based problems. It is a Python package built on PyTorch, JAX, and TensorFlow, making it easy to simulate systems with initial conditions. PhiFlow supports both Eulerian (grid-based) and Lagrangian (particle-based) simulations. My focus is specifically on Eulerian simulations, and I tested PhiFlow using a volleyball float serve simulation as a proof of concept to visualize the turbulent behavior of a projectile's wake. Since air is a gas, PhiFlow can also be applied to aerodynamics, allowing modifications to properties like viscosity and pressure gradients to model different environments.

**Description**

PhiFlow works in conjunction with the Navier-Stokes equation to model fluid flow. The general process involves defining a domain (grid resolution, size, and boundaries), initializing vector fields (velocity, pressure, smoke), and applying forces (gravity, inflow, obstacles like walls, or external forces). Solving the Navier-Stokes equation establishes the relationship between a particle's momentum and factors such as pressure gradients, viscosity, and external forces.

Users can also include advection and diffusion to simulate fluid movement under the constraints of Navier-Stokes. Diffusion controls the outward distribution of density per time step at a user-defined rate, while advection moves smoke along velocity vectors through the domain. PhiFlow provides several methods for calculating the next state, including Semi-Lagrangian and MacCormack schemes. While MacCormack is more accurate, it has less diffusion and can overshoot.

**Applications**

PhiFlow has numerous applications in differential physics, including aerodynamics, ocean and tsunami simulations, blood flow modeling, and buoyancy-driven flow. In aerodynamics, it can simulate airflow over wings and vehicles. In medical applications, it is useful for modeling vascular fluid dynamics.

For my volleyball simulation, I initialized a standard velocity and smoke field, created a SoftGeometryMask object representing the volleyball with zero velocity, and added inflow spawns with rightward acceleration to model air flowing against the ball. Running the simulation produced turbulent behavior. PhiFlow likely has potential for more advanced volleyball simulations beyond this proof of concept.

Additionally, I discovered why the "walls" of the volleyball did not absorb the vector flow. This was due to the no-slip condition in the code, which assumes that at the fluid-solid interface, there is no tangential component of relative velocity—meaning the velocity at the wall itself is zero.

**References**

YT: Machine Learning & Simulation
https://www.youtube.com/watch?v=KMfcF9XvVio&list=PLYLhRkuWBmZ5R6hYzusA2JBIUPFEE755O&index=5
This YT series from the creator of PhiFlow himself:
https://www.youtube.com/watch?v=YRi_c0v3HKs&list=PLYLhRkuWBmZ5R6hYzusA2JBIUPFEE755O&index=2
But how DO Fluid Simulations Work?
https://www.youtube.com/watch?v=qsYE1wMEMPA
DeepSeek LLM for helping with code